WEST Search History

DATE: Thursday, April 03, 2003

Set Name side by side	Query	Hit Count	Set Name result set
	DB=USPT; PLUR=YES; OP=ADJ		
L24	L22 and l6	1	L24
L23	L22 and l9	0	L23
L22	cookie same domain same field	9	L22
L21	cookie and domain and L20	1	L21
L20	5983208.pn.	1	L20
L19	L18	14	L19
L18	kerberos and L17	14	L18
L17	L16 and cookie and domain	14	L17
L16	l12 and L15	14	L16
L15	l1 and l2	74	L15
L14	cookie and domain and L13	0	L14
L13	6136772.pn.	1	L13
L12	L11 and kerberos	14	L12
L11	L10 and digital	24	L11
L10	L9 and algorithm	24	L10
L9	L8 and encrypt$4 and signature$2	24	L9
L8	L7 and (password$2 or pass-word$)	30	L8
L7	L6 and ip and time	32	L7
L6	L5 and authenticat$4	47	L6
L5	L4 and l1	58	L5
L4	L2 and secur$4 and l2	6526	L4
L3	L2 and secur$4 and l2	6526	L3
L2	client and server	16251	L2
L1	cookie and domain and expirat$4	79	L1

END OF SEARCH HISTORY

WEST

End of Result Set

☐ Generate Collection Print

L24: Entry 1 of 1 File: USPT Jun 20, 2000

DOCUMENT-IDENTIFIER: US 6078956 A
TITLE: World wide web end user response time monitor

Abstract Text (1):
A method of logging information in a computer network having a Web client connectable to a Web server. In response to the HTTP request (and as a result of receiving a response to that request), a response time associated with that first HTTP request is calculated. Thereafter, the response time calculated is passed from the Web client on a subsequent HTTP request to that Web server, where the information is logged for later use. In a preferred embodiment, the response time associated with the first HTTP request is passed in a cookie of the second HTTP request.

Brief Summary Text (3):
The present invention relates generally to computer networks and more particularly to a method and system for monitoring and collecting data in a client-server computer network such as the Internet.

Brief Summary Text (5):
The World Wide Web is the Internet's multimedia information retrieval system. In the Web environment, client machines effect transactions to Web servers using the Hypertext Transfer Protocol (HTTP), which is a known application protocol providing users access to files (e.g., text, graphics, images, sound, video, etc.) using a standard page description language known as Hypertext Markup Language (HTML). HTML provides basic document formatting and allows the developer to specify "links" to other servers and files. In the Internet paradigm, a network path to a server is identified by a so-called Uniform Resource Locator (URL) having a special syntax for defining a network connection. Use of an HTML-compatible browser (e.g., Netscape Navigator or Microsoft Internet Explorer) at a client machine involves specification of a link via the URL. In response, the client makes a request to the server identified in the link and receives in return a document formatted according to HTML.

Brief Summary Text (6):
The time period between the issuing of an HTTP request from the browser and the return of the requested document (or some component thereof) is known as the end user "response time." The response time is a function of the time spent servicing the HTTP request at the Web server together with the network transmission time to and from the server. The network transmission time is sometimes referred to herein as the "Internet delay."

Brief Summary Text (7):
Response times could be measured in environments where the clocks of the client and server machines are synchronized or where an external time reference is available to both the client and the server. Given the diverse nature of the Internet environment, however, such solutions are impractical because these criteria could not be met for all of the clients of a large web server. One possible alternative would be to place a special client (a so-called "transaction monitor") on the Internet and have the transaction monitor periodically issue a request to the server of interest. The transaction monitor would have to be built with response time instrumentation applied. Measured response times for this client would then be

presumed to be indicative of response times that actual clients encountered. Such an approach also has significant drawbacks. The transaction monitor would presumably hook into the Internet at a fixed site or ISP (or at most a small number of sites). The response times measured by the transaction monitor would thus represent only a small fraction of the total paths that may be used to connect to a large server. In addition, such a transaction monitor would be useless for resolving questions about the response times for requests issued by way of an ISP that the transaction monitor is not currently using. Further, the transaction monitor would have to be constructed to use test transactions against the server. Test transactions are suspect in that they may omit entire classes of operations, and they can be difficult to create if the mission of the Web server is considered critical or secure (e.g., financial transactions).

Brief Summary Text (10):
It is thus a primary object of the present invention to measure response times as seen by an end user for requests submitted from a Web browser to a Web server on the World Wide Web.

Brief Summary Text (11):
It is another primary object of this invention to measure end user response times at the Web browser and to submit such data to a Web server for

Brief Summary Text (13):
It is another important object of this invention to implement end user response time monitoring capability without resort to synchronized clocks at the client and server machines, to use of an external time reference, or to implementation of a dedicated transaction monitor.

Brief Summary Text (16):
These and other objects of the invention are achieved by calculating a response time associated with a first HTTP request and then passing that response time information from the Web client to the Web server in a second, subsequent HTTP request. Preferably, the response time information is passed to the Web server in a special cookie called a response time protocol (RSP) cookie.

Brief Summary Text (17):
Thus, in one preferred embodiment of this invention, a method of logging information in a computer network having a Web client connectable to a Web server comprises several steps. The method begins at the Web client in response to a first HTTP request. In response to the HTTP request (and as a result of receiving a response to that request), a response time associated with that first HTTP request is calculated. Thereafter, the response time calculated is passed from the Web client on a subsequent HTTP request to that Web server, where the information is logged for later use (e.g., URL statistical analysis and the like). In a preferred embodiment, the response time associated with the first HTTP request is passed in a cookie of the second HTTP request.

Brief Summary Text (18):
The response time associated with the first HTTP request is calculated without synchronized client and server machine clocks or use of an external timebase. Rather, the end user response time is calculated using just a clock in the Web client. In particular, this calculation begins by recording a first clock time on the clock, the first time associated with the transmission of the first HTTP request from the Web client to the Web server. Upon receipt at the Web client of a specified event in a response to the first HTTP request, a second clock time on the clock is recorded. The response time is then generated by subtracting the first clock time from the second clock time. The specified event in the response to the first HTTP request may be any event in the response, such as first packet return, last HTML byte, last .gif byte, receipt of some intermediate byte or page element, or the like (provided this is done consistently from request to request). Because the "timebase" for the response time calculation is all "local" (i.e. within the Web client itself), any given event may be used as the trigger for the calculation. The calculated response time is saved and then sent to the Web server upon a next HTTP request to that server.

Brief Summary Text (19):
Preferably, the present invention is implemented with a client "piece" and a server "piece." The client software may be a computer program product implemented in a computer-readable medium or otherwise downloaded to the Web client over the computer network. This software includes first program code means, responsive to a first HTTP request from the Web client to the Web server, for calculating the response time associated with the first HTTP request, and second program code means, responsive to the calculating means and a second HTTP request, for placing the response time in a cookie associated with the second HTTP request. The second HTTP request is then sent to the Web server to pass the response time information as previously described.

Brief Summary Text (20):
The server "piece" of the invention is also preferably implemented in software associated with a Web server program running on a Web site that supports the invention. Such a Web site is sometimes referred to herein as an "instrumented" server. The server "piece" includes a first program code means, responsive to receipt of a current HTTP request from the Web client, for retrieving a cookie from the HTTP request. The cookie includes information identifying a response time associated with a prior HTTP request from the Web client to the Web server program. The server piece also includes second program code means responsive to the retrieving means for logging the response time.

Drawing Description Text (4):
FIG. 2 is a flowchart illustrating the conventional processing associated with an HTTP request from the Web client to the Web server shown in FIG. 1;

Drawing Description Text (7):
FIG. 5 is a flowchart illustrating the various operations of the Web client and the Web server in response to a user HTTP request.

Detailed Description Text (2):
A representative system in which the present invention is implemented is illustrated in FIG. 1. A client machine 10 is connected to a Web server platform 12 via a communication channel 14. For illustrative purposes, channel 14 is the Internet, an Intranet or other known network connection. Web server platform 12 is one of a plurality of servers which are accessible by clients, one of which is illustrated by machine 10. A representative client machine includes a browser 16, which is a known software tool used to access the servers of the network. Representative browsers include, among others, Netscape Navigator (Version 2.0 and higher), Microsoft Internet Explorer (Version 3.0 and higher) or the like, each of which are "off-the-shelf" or downloadable software programs. The Web server platform (sometimes referred to as a "Web" site) supports files in the form of hypertext documents and objects. In the Internet paradigm, a network path to a server is identified by a so-called Uniform Resource Locator (URL). The World Wide Web is the Internet's multimedia information retrieval system. In particular, it is a collection of servers of the Internet that use the Hypertext Transfer Protocol (HTTP), which provides users access to files using Hypertext Markup Language (HTML).

Detailed Description Text (3):
A representative Web Server platform 12 comprises an IBM RISC System/6000 computer 18 (a reduced instruction set of so-called RISC-based workstation) running the AIX (Advanced Interactive Executive Version 4.1 and above) Operating System 20 and a Web server program 22, such as Netscape Enterprise Server Version 2.0, that supports interface extensions. The platform 12 also includes a graphical user interface (GUI) 24 for management and administration. The various models of the RISC-based computers are described in many publications of the IBM Corporation, for example, RISC System/6000, 7013 and 7016 POWERstation and POWERserver Hardware Technical Reference, Order No. SA23-2644-00. AIX OS is described in AIX Operating System Technical Reference, published by IBM Corporation, First Edition (November 1985), and other publications. While the above platform is useful, any other suitable hardware/operating system/Web server combinations may be used.

Detailed Description Text (4):
The Web Server accepts a client request and returns a response. The operation of the

server program 22 is governed by a number of server application functions (SAFs), each of which is configured to execute in a certain step of a sequence. This sequence, illustrated in FIG. 2, begins with authorization translation (AuthTrans) 30, during which the server translates any authorization information sent by the client into a user and a group. If necessary, the AuthTrans step may decode a message to get the actual client request. At step 32, called name translation (NameTrans), the URL associated with the request may be kept intact or it can be translated into a system-dependent file name, a redirection URL or a mirror site URL. At step 34, called path checks (PathCheck), the server performs various tests on the resulting path to ensure that the given client may retrieve the document. At step 36, sometimes referred to as object types (ObjectType), MIME (Multipurpose Internet Mail Extension) type information (e.g., text/html, image/gif, etc.) for the given document is identified. At step 38, called Service (Service), the Web server routine selects an internal server function to send the result back to the client. This function can run the normal server service routine (to return a file), some other server function (such as a program to return a custom document) or a CGI program. At step 40, called Add Log (AddLog), information about the transaction is recorded. At step 42, called Error, the server responds to the client when it encounters an error. Further details of these operations may be found in the Netscape Web Server Programmer's Guide, Chapter 5, which is incorporated herein by reference. The time spent carrying out the various functions of FIG. 2 (namely, the time spent by the Web server serving the HTTP request) is referred to below as the SERVER.sub.-- PROCESSING.sub.-- TIME.

Detailed Description Text (5):
Thus, the Web server 18 includes a known set of server application functions (SAFs). These functions take the client's request and other configuration data of the server as input and return a response to the server as output. Referring back to FIG. 1, the Web server 18 also includes an Application Programming Interface (API) 26 that provides extensions to enable application developers to extend and/or customize the core functionality thereof (namely, the SAFs) through software programs commonly referred to as "plug-ins."

Detailed Description Text (6):
FIG. 3 illustrates the various components that comprise the "response time" of a given HTTP request. A first time "x" represents the time to transfer the HTTP request (usually a GET or POST request) from the Web client to the Web server. A second time "y" represents the server processing time (SERVER.sub.-- PROCESSING.sub.-- TIME), which was described above with respect to the flowchart of FIG. 2. A third time "z" then represents the time to transfer a specified event in the response to the HTTP request back to the Web client. The specified event may be first packet return, last HTML byte, last .gif byte, or some intermediate event within the transfer. Thus, the response time "r" equals "x+y+z." The "Internet delay," which is that portion of the response time associated with the actual transmission over the Internet itself (as opposed to the time spent within the server for processing), is then the value "x+z". The present invention provides a technique for calculating response time associated with a given HTTP request and then recording or logging that response time at the Web server for subsequent analysis and use.

Detailed Description Text (7):
A preferred embodiment of the inventive method is illustrated in the simplified flowchart of FIG. 4. The method begins at step 50. At step 52, a test is run repeatedly to determine whether a given HTTP request (e.g., a GET or POST) has been issued from the client to a given server. If the outcome of the test at step 52 is positive, the routine continues at step 54 to calculate the response time "x+y+z." A preferred technique for performing this calculation is described below. At step 56, the calculated response time is saved at the client. A test is then run at step 58 to determine if a new HTTP request for the given server. If the result of the test at step 58 is negative, the routine cycles and repeatedly tests for this condition. If, however, the result of the test at step 58 is positive, the routine continues at step 59 to retrieve the end user response time calculated at step 54 (which will usually be the response time associated with the most-recent visit to the server), and formulates a response time protocol (RSP) cookie at step 60. At step 62, the cookie is then passed to the Web server, preferably within the new HTTP request

itself. This completes the basic processing.

Detailed Description Text (8):
Thus, in accordance with the preferred embodiment, a response time associated with a first HTTP request is calculated as the HTTP request is being processed but not passed to the Web server until a subsequent HTTP request (typically, a next request). Preferably, the response time information is passed to the Web server in a special cookie associated with the subsequent HTTP request. A more detailed description of this process is now provided.

Detailed Description Text (9):
In a preferred embodiment, a special "response time cookie" (herein referred to as the "RSP cookie") is associated with the Web client browser and processed by the Web server. The RSP cookie comprises the following two fields of data (and possibly others, as will be described):

Detailed Description Text (12):
The particular format of LASTRSPTIME is not significant; however, for this example, it is assumed to be an integer representing the number of milliseconds required for the response. Web servers are divided into two classes: instrumented and non-instrumented. Instrumented servers are servers that implement the RSP cookie protocol. It is presumed that there is some mechanism used to inform the browser as to whether or not a particular server is an instrumented server. The default is for a Web server to be non-instrumented. Web client browsers that support the RSP cookie protocol are called instrumented browsers. For the rest of this discussion, it is assumed that the HTTP request originates from an instrumented browser.

Detailed Description Text (13):
The protocol then operates as follows. Whenever an instrumented server is contacted, the browser examines the cookie cache looking for a RSP cookie for that server. This is step 70. A test is then made at step 72 to determine if an RSP cookie is found. If no RSP cookie is found, the routine continues at step 74 with the browser constructing a new RSP cookie for that server by initializing LASTRSPTIME to -1 and LASTURL to null. If an RSP cookie is found (i.e. the output of the test at step 72 is positive), or after step 74, the browser includes the current contents of the RSP cookie for that server by inserting a "Cookie:" request into the header of the HTTP request and filling in the values for LASTRSPTIME and LASTURL from the cookie cache. This is step 76. At step 78, the browser records the time when the request is sent according to the time clock on the client system where the browser is running and stores this value in storage local to the browser (or in some other way accessible to the browser).

Detailed Description Text (14):
The routine then continues with the server processing. In particular, when the cookie (associated with the HTTP request) is received, the routine continues at step 80 and examines the LASTRSPTIME field. A test is then made at step 82 to determine if this field is negative. If so, the server processing ends, as indicated by step 84. If the outcome of the test at step 82 indicates that the LASTRSPTIME field is non-negative, the routine continues at step 86 with the server logging the LASTRSPTIME field and the LASTURL and any other information from the request the server might find appropriate (e.g. the domain name of the requester). To minimize the total amount of data collected, the server may choose to randomly sample the recorded response times and log only a subset of the times. Postmortem analysis tools would then examine the response time log and produce response time statistics per URL.

Detailed Description Text (15):
Processing then continues back at the client. In particular, at step 88, at test is made to determine whether a response has been received from an instrumented server. If the outcome of the test at step 88 is negative, the step cycles and continues to test. If, however, the outcome of the test at step 88 indicates that a response to the HTTP request is received from an instrumented server, the routine continues at step 90. In this

Detailed Description Text (16):

step, the browser subtracts from the current client clock time the time that the request was sent. This time, as well as the "referring URL," are recorded in the RSP cookie for that server at step 92. This completes the processing.

Detailed Description Text (17):
It should be noted that when the cookie cache is searched for a RSP cookie, the normal path matching of the Netscape cookie protocol is disabled. Equivalently, all RSP cookies are preferably stored with a path of "/". The effect of this is that the RSP cookie is sent as part of every request to an instrumented server. There is potentially a separate RSP cookie for each instrumented server that the Web browser has contacted, subject to the limits of storage on the client machine. Well known methods, such as expiration dates, can be used to discard RSP cookies when they are likely to no longer be useful. A discarded RSP cookie means that a response time observation has been lost. Since RSP time statistics are likely to be sampled by the Web server in any case, however, the loss of a response time observation due to this event is not considered harmful.

Detailed Description Text (18):
This method does not return the response time of the last interaction the browser has with a server during a particular session since the response time of the previous interaction is always sent out with the next request. However, if sufficient space is available on the browser system, and if the RSP cookie is not discarded before the next time that the browser visits the instrumented server, the response time of the last request of the previous session may be presented to the server as part of the first request of the new session.

Detailed Description Text (19):
Naming conventions for the RSP Cookie fields LASTRSPTIME and LASTURL may vary depending on the conventions observed by the Web server. For example, to communicate these values to a server-side JavaScript program running on a Netscape Enterprise server, these fields could be named (respectively):

Detailed Description Text (22):
As also noted above, the present invention uses a special "cookie." Those of ordinary skill in the art will appreciate that this RSP cookie is a variant of the persistent client state HTTP cookie normally used to facilitate authentication of a user connecting to an enabled web site. Cookies are a known Internet mechanism which server-side connections (such as CGI scripts) can use to both store and retrieve information on the client side of the connection. A server, when returning an HTTP object to a client, may also send a piece of state information which the client will store. Typically, the state object, called a "cookie," may include a description of the range of URLs for which that state is valid. According to the Persistent Client State HTTP Cookies Preliminary Specification, which is hereby incorporated by reference and may be viewed at netscape.com at the path "/newref/std/cookie.sub.--spec.sub.-- html," a cookie is introduced to the client by including a Set-Cookie header as part of an HTTP response, usually through a CGI script.

Detailed Description Text (23):
It should be appreciated that the "response time" is not necessarily the time between the initiation of the HTTP request at the client and the first or last packet return associated with the target document. This invention could also be used to record and collect intermediate response times such as: response time of first data arrival, response time HTML delivery complete, response time of all gift's delivered, or some other intermediate response time. If the domain name of the requester is logged as part of the server log record, one could also use this tool to determine whether poor response times are due to a particular ISP or other route into the server.

Detailed Description Text (24):
This invention solves the problem of measuring response times as seen by the end user for requests submitted from a Web browser to a Web server on the World Wide Web. The invention is suitable for use in a production environment where users submit requests over the Internet either by direct connection or by connection through an Internet Service Provider over dial up lines. Response times are measured by the Web browser and submitted to the Web server for collection.

Detailed Description Text (25):
According to the invention, it may be desirable to implement the end user response time monitor using client and server side JavaScript. Familiarity with basic JavaScript programming conventions is assumed in the following discussion.

Detailed Description Text (26):
As discussed above, one aspect of the present invention involves making a local timestamp when a request is made to the Web server. This operation may be performed using a simple "javascript" URL. For example, consider the following conventional anchor and link:

Detailed Description Text (30):
where the function rspmon is a client-side JavaScript function defined as follows:

Detailed Description Text (31):
The first line of this function records the current clock time in the cookie so that this value will be available as part of the "client" object in the server side JavaScript. The second line of the function causes the link to be followed. This function is then included on every page that includes a javascript:rspmon() URL. To properly measure the response time for accessing URL foo, all URL's referencing foo are changed to javascript:rspmon(foo)".

Detailed Description Text (32):
In this example, when the user clicks on the "click here, big spender" link, the rspmon() function is invoked. The function records the current client system clock time and causes the link to be followed. When the request is sent up to the server, the cookie values are sent along with the request so the server now has the client submission time given in milliseconds since Jan. 1, 1970 (which is the date that getTime() returns) according to the client system's clock value.

Detailed Description Text (33):
To calculate the response time necessary to serve up "www.bigbank.url", the following code is included in the server-side JavaScript for that URL:

Detailed Description Text (34):
The above code is included so that the client-side JavaScript it emits is included in the "onload" event handler action for the page. The first statement sets a flag in the cookie to make sure that last.sub.-- rsp.sub.-- time is updated on the client only on the first time that the onload action for the page is executed. (The onload action is executed each time the page is reloaded or resized). The second statement sets a value in the cookie (here set by some pseudocode) that includes trace information to allow correlation of this response time with other log records related to this request. The if.sub.-- trace.sub.-- id variable is intended to return the current IFS trace id; the trans.sub.-- count is intended to return a count of transactions during the session.

Detailed Description Text (35):
If NETSCAPE.sub.-- LIVEWIRE.curr.sub.-- time on the client was set to 987666532, then the rest of the statements in the code above cause the following <A name=h227 JavaScript to be submitted to the <A name=h228 href=#h229>client</A> as part of the

<U>Detailed Description Text</U> (36):
The if statement ensures that last.sub.-- rsp.sub.-- time is updated only the first time that the "onload" action is invoked. The last.sub.-- rsp time contains the time in milliseconds required to service this request and deliver the page to the <A name=h229 href=#h230

<U>Detailed Description Text</U> (37):
To collect response time data, code is added on the <A name=h230 href=#h231>server</A> that inspects the <A n last.sub.-- rsp time and rsp.sub.-- time.sub.-- set. If rsp.sub.-- time.sub.-- set has the value RSP.sub.-- TIME.sub.-- SET, then client.last.sub.-- rsp.sub.-- time and client.rsp.sub.-- time.sub.-- trace.sub.-- id are logged. Client.rsp.sub.-- time.sub.-- set is then set to RSP.sub.-- TIME.sub.-- NOT.sub.-- SET (on the <A name=h232 href=#h233>server</A>). The latter operation is the same response time more than once and to deal with mixtures of instrumented and non-instrumented pages.

<U>Detailed Description Text</U> (38):
Thus, in this approach, each

URL specifying the target page is a JavaScript URL of the form javascript:rspmon(), and each <A name=h233 href=#h234>client</A> page includes the r described above. Target page instrumentation includes the <A name=h234 href=#h235>s JavaScript that outputs the onload action to record the last.sub.-- rsp.sub.-- time and update resp.sub.-- time.sub.-- set in the <A name=h235 href=#h236>cookie</A

<U>Detailed Description Text</U> (41):
As used herein, "Internet<A name=h client</A>" should be broadly construed to mean any computer or component thereof directly or indirectly connected or connectable in any known or later-developed manner to a computer network, such as the Internet. The term "Internet <A name=h237 should also be broadly construed to mean a computer, computer platform, an adjunct to a computer or platform, or any component thereof. Of course, a "<A name=h238 href=#h239>client</A>" should be broadly construed to mean one who req and "<A name=h239 href=#h240>server</A>" is the entity which downloads the file.

<U>Detailed Description Text</U> (42):
The inventive concept of using a second, subsequent HTTP request (and preferably a <A name=h240 href=#h241>co therewith) as the "vehicle" for transporting information about a first, prior HTTP request may be used for other purposes as well as response time monitoring. Thus, it may be desirable to record information about other <A name=h24 activities (for example, relative to the first HTTP request, or some other<A name=h2 client</A> or browser action) and then use another HTTP request as the means by which that information is then delivered back to the <A name=h243 href=#h244>server< invention should be broadly construed as covering any method of logging information in a computer network having a Web <A name=h244 href=#h245>client</A> co server wherein a cookie</A> (in a current interaction) is used to transmit information about a previous <A name=h246 href=#h247>client-server</A> interaction.

<U>Detailed Description Text</U> (44):
Having thus described our invention, what we claim as new and desire to <A name=h247 href=#h248>secure</A> by forth in the following claims.

<U>Detailed Description Paragraph Table</U>

(4):
_____ <html> <head>
<title>Response Monitor <A name=h248 href=#h249>Client</A> Test Page</title> </head
<SCRIPT LANGUAGE="JavaScript"> // // this is PSEUDO code // there is no GLOBAL <A name=h249 href=#h250>cookie</A> in current JavaScript // what is needed i into the cookie</A> cache // with "path=/". At present, <A name=h251 href=#h252>coo // "path=current.sub.-- url" via way of the document.cookie attribute // // the GLOBAL attribute is needed to make sure the <A name=h252 href=#h253>cookie</A> i server</A> regardless of the target url // function rspmon (url) { var now = new Date (); // // use the GLOBAL <A name=h254 href=#h255>cookie</A> (assumed js extens GLOBAL.cookie = "NETSCAPE.sub.-- LIVEWIRE.currtime="+now.getTime()+";"; GLOBAL.cookie = "NETSCAPE.sub.-- LIVEWIRE.currurl ="+url+";"; window.location = url; // // It is assumed the default action of the <A name=h255 href=#h256>server all global <A name=h256 href=#h257>cookies</A> back to the <A name=h257 href=#h258>c is required on the <A name=h258 href=#h259>server</A> side for this request // </SC [End User Response Time Monitor Sample <A name=h259 href=#h260>Client</A>

 [<A HREF="index.html">Home Page </A> .vertline. <A HREF="javascript:rspmon (`http://www.webserver.example.com/rspmontarget`) ">Test</A>
</body> </html> </html> <head> <title>Test response time monitor target page</title> </head> <body>
 [This page is a response time monitor target page]

 <SCRIPT> // // this is PSEUDO code // there is no GLOBAL <A name=h260 href=#h261>cookie</A> in current JavaScript // what a <A name=h261 href=#h262>cookie into the cookie</A> cache // with "path=/". At pre inserted with // "path=current.sub.-- url" via way of the document.cookie attribute // // the GLOBAL attribute is needed to make sure the <A name=h263 href=# sent to the <A name=h264 href=#h265>server</A> regardless of the target url // // / code in Netscape JavaScript Reference // function getCookie(Name) { var search = "NETSCAPE.sub.-- LIVEWIRE."+Name+"="; var RetStr = ""; var offset = 0; var end = 0; if (GLOBAL.cookie.length > 0) { offset = GLOBAL.cookie.indexOf (search); if (offset != -1) { offset += search.length; end = GLOBAL.cookie.indexOf(";",offset); if (end == -1) end = document.cookie.length; // // Note well: It is assumed here that there are no // special characters in the value!

<U>Detailed Description Paragraph Table</U> (5):
// RetStr =

```
GLOBAL.cookie.substring(offset,end);  } } // document.write("<P>At
return. . .RetStr="+RetStr+".backslash.n");  return(RetStr);  } // // code
based on sample code in Netscape JavaScript Reference  // function
setCookie(Name, Value) {  // // Note well: no special characters are allowed in
the  value!  // GLOBAL.cookie = "NETSCAPE.sub.-- LIVEWIRE."+Name+"="
+Value+";";  } function expireCookie(Name) {  // // make this name disappear
from the <A name=h265 href=#h266>cookie</A>  // var now = new Date ();  // choose an
seconds in the past  now.setTime(now.getTime() - 10000);  GLOBAL.cookie =
"NETSCAPE.sub.-- LIVEWIRE."+Name+  "=EXPIRED;  expires="+  now.toGMTString();
} // // calculate the response time of the last request based on the  <A name=h267 h
values sent up the the <A name=h268 href=#h269>server</A> at that time  // function
currtime = getCookie("currtime");  // // did we get a currtime sent to us in
the <A name=h269 href=#h270>cookie</A> from the  <A name=h270 href=#h271>server</A>
the <A name=h271 href=#h272>cookie</A> from getting cluttered  // expireCookie ("las
expireCookie ("lasturl");  var now = new Date();  setCookie
("lastrsptime",now.getTime() -currtime);  // // get the url of the last request
-- it was put in  // the <A name=h272 href=#h273>cookie</A> as currurl when the last
setCookie ("lasturl",getCookie ("currurl"));  } // // we clear currtime here so
that we won't recalculate the  lastrsptime  // if the page gets reloaded before
a new request is made of  the <A name=h273 href=#h274>server</A>  // expireCookie ("
expireCookie ("currurl");  } // // It is assumed the default action of the<A name=h2
server</A> is to return  // all global <A name=h275 href=#h276>cookies</A> back to t
request so  // that no action is required on the <A name=h277 href=#h278>server</A>
// </SCRIPT>  </SCRIPT>  // // since this is response time monitor
target page,  // the <A name=h278 href=#h279>cookie</A> contains the currtime and cu
request. we can use these to calculate  // the response time of the last
request.  // // go do that now  // updateRspTime();  </SCRIPT>
<BR><BR><BR><BR>  [ <A HREF="index.html">Home
Page</A> .vertline.  <A
HREF="http://www.webserver.example.com/rspmonfinal"  >Final Page</A>
]  </body>  </html>  </html>  <head>
<title>Response Time Monitor Final Page</title>  </head>
<body>  <center>  [This page includes the code required to log a
response time]  <BR><BR><BR>  <SERVER>  logfile = new
File ("/usr/ns-home/LiveWire/samples/testjst/rsptime.log");  now = new Date();
logfile.writeln("now="+now+". . ."+client.lastrsptime+"  url:"+client.lasturl);
logfile.close();
```

Other Reference Publication (3):
Giacone, Glynn B.; "Seek and fine-tune: Getting the most from client-server transactions"; Data Based Advisor, Data Based Solutions Inc.; v14, n9, p76(7), Sep. 1996.

CLAIMS:

1. A method of logging information in a computer network having a Web client connectable to a Web server, comprising the steps of:

calculating a response time associated with a first HTTP request from the Web client to the Web server; and

passing the response time associated with the first HTTP request using a second HTTP request from the Web client to the Web server;

wherein the second HTTP request includes a response time protocol cookie in which the response time associated with the first HTTP request is passed.

4. The method as described in claim 1 wherein the response time associated with the first HTTP request is calculated using a clock in the Web client.

5. The method as described in claim 4 wherein the response time calculation includes the steps of:

recording a first clock time on the clock, the first time associated with the transmission of the first HTTP request from the Web client to the Web server;

determining a second clock time on the clock, the second time associated with receipt at the Web client of a specified event in a response to the first HTTP request; and

subtracting the first clock time from the second clock time to generate the response time.

7. A method of logging information in a computer network having a Web client connectable to a Web server, comprising the steps of:

at the Web client, calculating a response time associated with a first HTTP request from the Web client to the Web server;

passing the response time associated with the first HTTP request in a response time protocol cookie of a second HTTP request from the Web client to the Web server; and

at the Web server, recording the response time associated with the first HTTP request.

9. The method as described in claim 7 wherein the response time associated with the first HTTP request is calculated using a clock in the Web client.

10. The method as described in claim 9 wherein the response time calculation includes the steps of:

recording a first clock time on the clock, the first time associated with the transmission of the first HTTP request from the Web client to the Web server;

determining a second clock time on the clock, the second time associated with receipt at the Web client of a specified event in a response to the first HTTP request; and

subtracting the first clock time from the second clock time to generate the response time.

12. A computer program product in a computer-readable medium for monitoring response time in a Web client connectable to a Web server in a computer network, comprising:

first program code means, responsive to a first HTTP request from the Web client to the Web server, for calculating a response time associated with the first HTTP request; and

second program code means, responsive to the calculating means and a second HTTP request, for placing the response time in a response time protocol cookie associated with the second HTTP request.

13. The computer program product as described in claim 12 wherein the Web client includes a clock and the calculating means comprises:

means for recording a first clock time on the clock, the first time associated with the transmission of the first HTTP request from the Web client to the Web server;

means for determining a second clock time on the clock, the second time associated with receipt at the

Web client of a specified event in a response to the first HTTP request; and

means for subtracting the first clock time from the second clock time to generate the response time.

14. A computer for use as a server in a computer network having a Web client connectable to the computer, comprising:

a processor;

an operating system;

a Web server program;

first program code means responsive to receipt of a current HTTP request from the Web client for retrieving from the HTTP request a response time protocol cookie, the response time protocol cookie including information identifying a response time associated with a prior HTTP request from the Web client to the Web server program; and

second program code means for logging the response time.

15. A method of logging information in a computer network having a Web client connectable to a Web server, comprising the steps of:

calculating information associated with a first HTTP request from the Web client to the Web server; and

passing the calculated information in a response time protocol cookie of a second HTTP request from the Web client to the Web server.

17. A method of logging information in a computer network having a client connectable to a server, comprising the steps of:

calculating a response time associated with a first transfer protocol request from the client to the server; and

passing the response time associated with the first transfer protocol request using a second transfer protocol request from the client to the server;

wherein the second transfer protocol request includes a response time protocol cookie in which the response time associated with the first transfer protocol request is passed.

First Hit Fwd Refs
End of Result Set

☐ Generate Collection Print

L3: Entry 1 of 1 File: USPT Oct 5, 1999

DOCUMENT-IDENTIFIER: US 5963915 A
TITLE: Secure, convenient and efficient system and method of performing trans-internet purchase transactions

Brief Summary Text (3):
The present invention is generally related to systems of performing commercial activities over a general access computer network and, in particular, to a system and method of conveniently and efficiently performing advertising responsive secure commercial purchase transactions over the Internet utilizing the World Wide Web.

Brief Summary Text (6):
One of the anticipated uses of the Web has been to provide a venue for commercial transactions in products and services. However, commercial use of the Web has distinctly not met the anticipated potential for a number of reasons. These reasons include security, convenience of use, and efficiency. Regarding security, current conventional Web browsers generally provide for the use of a reasonably secure encryption protocol overlaid on the HTTP protocol. The encryption protocol, typically involving a key-exchange based encryption algorithm, permits individual transactions over the Internet to be secure. Consequently, sensitive information, such as credit card numbers and the like, can be reasonably transferred over the Internet with little risk that the information can be misappropriated and misused.

Brief Summary Text (7):
An exemplary security system utilized by conventional HTTP browsers and servers is known as the secure sockets layer (SSL). The secure sockets layer defines and implements a protocol for providing data security layered under various application protocols, such as HTTP in particular, and over a conventional TCP/IP communications stack. The secure sockets layer protocol discretely provides the potential for data encryption, server authentication, message integrity, and client authentication for supported protocol connections over a TCP/IP connection. In use, the secure sockets layer is implemented at both the client browser and server ends of a network connection. A conventional uniform resource locator (URL), utilizing "https" as the secure HTTP protocol identifier, is issued by the client browser to specifically request a secure client/server session. A series of handshake transactions are provided to negotiate the establishment of the secure session including performing an encryption key exchange that is used in an encryption algorithm implemented by both the client-side and server-side secure sockets layers.

Brief Summary Text (8):
As part of this handshaking, the client browser may also retrieve the authentication certificate of the server for validation against a known certificate authority to ensure that the server is not an imposter. The secure HTTP protocol permits the server to also request and validate the authentication certificate, if any, held by the client. However, in general, client browsers and, more specifically, their client host computer systems are rarely registered with a publicly accessible authentication certificate authority. Thus, general use of client certificate authentication is not a viable means for identifying specific

client users or client computer systems.

Brief Summary Text (9):
As a consequence, commercial use of the Web to sell products and services practically requires the establishment of a forms based user identification scheme, typically based on user name and password, by the server system to securely identify and re-identify a specific client user. Providing the user name and password to initiate each purchase session with a particular server is the minimum required to authenticate the client user. The underlying secure HTTP protocol session ensures that the user name and password are securely transmitted in an encrypted form over the Internet to the correct server. By the fundamental nature of the key exchange encryption algorithm used, only the server can decrypt to clear text the user name and password provided from the client browser.

Brief Summary Text (10):
A secure HTTP session may span a number of individual HTTP transactions between a client browser and server. With each of these individual transactions, the exchange keys are in effect permuted synchronously by both the browser and server to vary the encryption coding used for each transaction. However, each established secure HTTP session requires definite closure to prevent a security breach commonly known as a "third party assumption of identity attack." That is, a third party may be able to continue the secure session started by another client browser relative to the server. Since client user authentication only occurs at the initiation of the secure session, the third party fully assumes the authorization of the session initiating client browser.

Brief Summary Text (11):
Consequently, commercial transactions over the Internet conventionally requires three distinct phases in order to securely perform a purchase transaction. The first phase, conducted once a secure HTTP session is established, is a logon transaction where the client user provides a user name and password for authentication by the server. Once authenticated, a second or selection phase allows the client user to select products and services for purchase. The server system must in some way continually track and manage the selections made by the client. The server may record or log each selection against the client account as the selections are made. This second phase is therefore an extended transaction that is made up of many discrete HTTP transactions. Such an extended selection transaction is subject to failure for a variety of conventional reasons, including simply an extended delay in the selection process, resulting in an incomplete or incorrect record of partial purchase selections being kept by the server system. Without authoritative closure of the purchase transaction, the server system typically aborts the purchase and discards the record of selected items.

Brief Summary Text (12):
A facility known as persistent client-side cookies has been introduced to provide a way for server systems to store selected information on client systems. Cookies are created at the discretion of the server system in response to specific client URL requests. Part of the server response is a cookie consisting of a particularly formatted string of text including a cookie identifier, a cookie path, a server domain name and, optionally, an expiration date, and a secure marker. The cookie is automatically discarded by the client system based on the expiration date. If the secure marker is present, then the cookie is only returned to a server system during a secure transaction. Where a URL client request made by the client, the cookie paths and domain names of cookies stored by the client are compared with those of the URL request. Cookies with matching paths and domain names are passed with the client URL request to the server system. Any text associated with the identifier is also passed back to the server system. In Internet purchasing applications, the identifiers and associated text can be used to store information about the current purchase selections.

Brief Summary Text (13):
Finally, the third phase requires some action on part of the client user to initiate closure of the purchase transactions and secure session. Typically, the third phase is entered when the client user indicates that all product and service selections are complete. A summary order confirmation form is then presented by the server. The purchase transaction and the secure session are closed on acceptance or cancellation of the order as presented.

Brief Summary Text (14):
The convenience of conventional purchase transactions via the Internet, however, leaves much to be desired. Because of the security concerns, a secure purchase session is limited to encompassing a single vendor at a time due to the required three phase login, select, commit purchase protocol required to ensure the integrity of a secure purchase session. Not only is the three phase purchase transaction itself self-evidently cumbersome and thereby a limiting barrier to convenient use by client users, but many purchase may involve only a single purchasable item or some number of items that are available only from distinctly different vendors. Where the purchase transaction is only for a single item, the necessity of executing a complete three phase purchase protocol distinctly reduces the likelihood that a client user will actually bother to make the purchase. A greater barrier exists where the purchase transaction, from the client user's perspective, is of multiple items from multiple vendors. The necessity of completing independent three phase purchase transactions with each of the vendors, particularly where the purchased items are not entirely planned for or subject to comparative inter-dependant selection, presents a significant barrier to the client user conveniently and expediently making the purchase of products and services. The three phase purchase transaction is simply cumbersome and limiting and will become more so as products and services are more widely available from different vendors over the Web.

Brief Summary Text (16):
In addition, the conventional three phase purchase transaction greatly limits the flexibility of different types of vendors from being able to deliver ordered products and services in their chosen most efficient manner. All products selected during a secure purchase session are, in effect, ordered from the single vendor regardless of whether another vendor might actually be the source of a product or service sold. The server vendor receives the entire order and must independently place orders with supporting vendors by conventional means. As an implicit result electronic catalog vendors, agent vendors, and order-clearinghouse type vendors are constrained to separately processing each and every ordered product or service to lower-tier vendors. Although an electronic catalog vendor, for example, might wish to have each lower-tier vendor directly fulfill their part of an order, the server vendor must itself discriminate which products are to be sourced by which lower-tier vendor and provide shipping and indirect billing information. In general, direct order fulfillment from multiple vendors, though the purchase transaction appears to be simply with the electronic catalogue vendor, would be significantly more efficient for both the end user and the involved vendors. Products and services would be shipped or provided sooner and with less opportunity for error, while ordered products and services are automatically processed through to the correct vendor with the correct billing and shipping information.

Brief Summary Text (17):
Consequently, there is a clear need for the ability to perform purchase transactions over the Internet that are secure, convenient and efficient both for a client user and the many different vendors of products and services available over the Internet.

Brief Summary Text (19):
Thus, a general purpose of the present invention is to provide a method of efficiently performing secure purchase transactions over the Internet.

Brief Summary Text (21):
An advantage of the present invention is that essentially no redundant user input is required to stage and maintain a secure purchase transaction over the Internet.

Brief Summary Text (23):
A further advantage of the present invention is that secure purchase transactions can be implemented for unsecure servers of sponsored products and services.

Brief Summary Text (24):
Yet another advantage of the present invention is that server side automation can provides for automatic simultaneous purchase transaction handling for both secure and unsecure client browsers.

Detailed Description Text (4):
A "protocol.sub.-- identifier" of "http" specifies the conventional hyper-text transfer protocol. A URL request for a secure Internet transaction typically utilizes the secure protocol identifier "https," assuming that the client browser and Web server are presumed to support and implement the secure sockets layer. The "server.sub.-- path" is typically of the form "prefix.domain," where the prefix is typically "www" to designate a Web server and the "domain" is the standard Internet sub-domain.top-level-domain of the server system 16. The optional "web.sub.-- page.sub.-- path" is provided to specifically identify a particular hyper-text page maintained by the Web server.

Detailed Description Text (13):
Referring now to FIG. 2 a number of different scenarios are presented where the present invention is utilized in simple to complex purchase transactions that, at least from a client user's perspective, are all equally secure and convenient. Each, from the merchant vendor's perspective, is also quite efficient. In a first scenario, a Server-1 22 serves a Web page 24 to a client browser in response to a URL request/Web page service transaction T1. The Web page 24 may embed any number of hyperlinks including for example hyperlinks 26, 28, 30, 32. The hyperlink 26 may represent a direct reference to an embedded URL or an active image map that can be utilized to ultimately resolve a client user selection on a discrete portion of a displayed graphic to a specific URL. The image map representation can thus be utilized to provide multiple selectable choices regarding one or more products or services graphically depicted by the hyperlink 26. These different but related URLS preferably allow the client user to separately request further information about the indicated product or service, information regarding other related products and services, information regarding the availability, method of shipment and terms of purchase for the indicated product or service, and to directly issue a request to purchase the product or service.

Detailed Description Text (14):
Where the selection reflects a request for further information, an image map selection identifier is passed as part of a transaction T2 to a vendor Server-2 34. The Server-2 34 can be the same logical or physical server as Server-1 22 or a completely independent server. The requested information is returned to the client browser as part of the transaction T2 preferably in the form of a Web page. Where the image map selection is resolved to a request to purchase URL, the present invention preferably provides for the purchase URL to specify use of a secure HTTP session with the Server-2 34. In accordance with the secure protocol, such as implemented by the secure sockets layer, the Server-2 34 negotiates and establishes a secure session T2 with the client browser. Once the secure session is established, the purchase request URL is, at least in effect, issued to vendor Server-2 34. Any client-side stored cookie data that properly corresponds to the request URL is also passed to the Server-2 34. In the preferred instance where an authenticated credit relationship has been pre-established between the client user and the Server-2 34, the client-side cookie encodes information sufficient to re-

authenticate the client user to the Server-2 34. Where a client/vendor credit relationship has not been pre-established, a corresponding cookie will not exist on the client system 12. In this case, the Server-2 34 may initiate a conventional process of establishing and validating a credit relationship with the client user. Preferably, the Server-2 provides a registration form to the client browser for display and completion by the client user as part of the secure transaction T2. The registration form typically provides for the entry of a name, a password, a credit card number, billing and shipping addresses for the client user and possibly other relevant information. The resulting information is used by the Server-2 34, in accordance with the present invention, to create and store a client-side cookie on the client system 12 for use in connection with a subsequent URL purchase request. A database record is also preferably created in the database 36.

Detailed Description Text (15):
Cookie data, when received by the Server-2 34 in connection with a purchase request URL, is then used to lookup a client database record in the database 36. The cookie data may be decoded and compared with the record contents to validate the cookie. Assuming that the comparison is correct, the identified record is then used as the source of billing related information, needed by the Server-2 34 to fulfill the client user's purchase request.

Detailed Description Text (16):
When the decoded cookie information becomes available to the Server-2 34, directly as part of the secure transaction T2 or indirectly from form entered data, the Server-2 34 may then perform an automated credit extension authorization check to ensure that sufficient credit exists and may be extended to cover the present purchase transaction. Again assuming sufficient credit is available, the purchase of the product or service identified by the selected URL 26 can be presented to the client user for approval using a purchase confirmation form identifying the selected product or service, the billing and shipping related information, and provide active confirm and cancel buttons.

Detailed Description Text (18):
The data provided by the client user in response to the confirmation form is preferably returned to the Server-2 34 and may terminate the secure session portion of the HTTP transaction T2. The purchase of the product or service, if accepted by the client user, is then a sales order that can then be processed by the Server-2 34 or passed on to an automated order processing system for order fulfillment consistent with conventional order processing practices to provide for the delivery of the product or service purchased to the client user.

Detailed Description Text (19):
Thereafter, should the client user again select the purchase portion of the hyperlink image map 26, or any other purchase selection hyperlink that corresponds to the same vendor operating from the Server-2 34, a new secure session T2 is established, the client-side cookie is provided to the Server-2 34, and a confirmation form is presented to the client user. The client-side cookie provided during the secure session T2 specifically encodes sufficient information to authenticate the client user to the Server-2 34, thereby obviating the need for the client user to re-authenticate manually.

Detailed Description Text (21):
Selection of, for example, a direct hyperlink URL 28 on the Web page 24 results in the issuance of a URL request initiating a transaction T3 with a Server-3 38. Although, in this scenario, Server-3 38 is the direct sponsor of the URL 28 on the Web page 24 served by the Server-1 22, the Server-3 38 may not wish to or be able to participate in a secure purchase transaction. However, as the sponsor of the URL 28, Server-3 38 may require preemptive notification of the selection of the URL 28. Accordingly, the URL 28 may be provided on the Web page 24 as a redirection URL that identifies a Server-4 40 as the second URL within the redirection URL 28.

Where the Server-3 38 implements the secure sockets layer, the initial URL reference to the Server-3 38 may provide for a secure session within the transaction T3. Consequently, any and all data provided as part of the redirection URL for accounting purposes to the Server-3 38 is encrypted by operation of the secure transaction T3.

Detailed Description Text (22):
However, Server-3 38 need not and often will not be a secure server. The accounting data provided with the redirection URL will typically not be of a sensitive nature. Generally, the accounting data will provide an identification of the particular instance of the URL 28 that was selected by a client user. Furthermore, any client-side cookie corresponding to the initial URL or the redirection URL will also not contain any sensitive account or billing information relevant to the actual product or service identified by the URL 28. Rather, the cookie containing the sensitive information will be transferred only to the Server-4 40 within a secure session transaction T4.

Detailed Description Text (23):
The Server-3 38 returns a redirection message and the second URL to the client browser. As in the case of the purchase URL provided to the Server-2 34, the second URL of the redirection URL preferably specifies a secure protocol, such as "https," a specific Web server, such as Server-4 40, and includes a Web page path that can be used to identify a particular product or service. Thus, in response to the redirection message from Server-3 38, the client browser preferably autonomously operates to establish a secure session transaction T4 with the Server-4 40. The sensitive data provided by the client-side cookie that is selected specifically by the second URL of the redirection URL is therefore passed to the Server-4 40 within the secure session transaction T4. The secure purchase transaction T4 then completes in the same manner as described above with regard to transaction T2.

Detailed Description Text (25):
The use of the redirection URL allows any number of URL sponsoring Web servers to utilize the services of the Server-4 40. The support of redirected purchase transactions by Server-4 40, however, does not preclude the Server-4 40 from directly supporting purchase transactions, such as within a secure session transaction T5 initiated in response to the selection of a directly hyperlinked URL 30. Thus, the present invention provides substantial flexibility in the use of primary purchase transaction Web servers such as Server-4 40.

Detailed Description Text (27):
Selection of a URL 32 preferably results in the establishment of a transaction T6 with the Server-5 44. Although the Server-5 44 may be a secure server and preferably maintains a database 45 of client user account records and Web pages detailing certain products and services available for apparently direct purchase, the Server-5 44 may itself maintain pre-established credit relationships with any number of other servers, such as Server-4 40. In response to a URL request for product information or to purchase a selected product, the Server-5 44 may establish a transaction T7 with the Server-4 40. This transaction T7 may be an HTTP or possibly a non-HTTP transaction, such as a network filesystem read (NFS) or custom remote procedure call (RPC), that serves to retrieve responsive information from the Server-4 40, preferably as or suitable for incorporation in a Web page that is responsively served to the client user.

Detailed Description Text (28):
The transaction T7 may also represent a secure purchase transaction in accordance with the present invention. In this instance, the Server-5 44 acts as a product purchaser relative to the Server-4 40 and purchases based on a credit relationship pre-established between the Server-5 44 and the Server-4 40. The referenced account record held in the database 42 by Server-4 may be used to specifically identify the Server-5 44 as a possibly re-branding reseller of products purchased from Server-4

40. Accordingly, after so identifying Server-5 44, the Server-4 40 may issue an HTTP request to Server-5 44 to obtain the drop ship address of the ultimately purchasing client user and, potentially, the brand identification to be applied to the product upon shipping. Alternately, though perhaps less efficient, the purchased product can be shipped to Server-5 44 for reshipment to the client user.

Detailed Description Text (30):
In any event, where the Server-5 44, in response to a secure purchase transaction request T6, has identified the selected product or service as being available through Server-4 40, the Server-5 44 operates as a proxy for the client user and relays through a secure purchase transaction T7 the purchase request initiated by the client user. This will require the client user to have or to establish a credit relationship with the Server-4 40.

Detailed Description Text (31):
Alternately, the Server-5 44 may directly respond to the client user selection URL 32 by performing a secure purchase transaction T6. On confirmation of the purchase selection by the client user, the Server-5 44 may initiate an independent purchase transaction T7 with the Server-4 40. Consequently, the full range of flexibility in the utilization of the present invention is maintained in scenarios involving the interposition of the Server-5 44 in the secure purchase transaction path. Notably, in essentially all instances, the entire purchase transaction from the client user perspective remains a simple two phase operation requiring at most only two mouse clicks for complete execution of the purchase.

Detailed Description Text (32):
FIG. 3 provides a detailed flow diagram illustrating the operation of the present invention in the purchase of a product over the Internet 14. Initially, the client browser receives a Web page 50 having any number of embedded URLs, either directly or through an indirect reference provided by an image map. The client user selects 52 a product from the Web page for purchase. The client browser ascertains 54 the corresponding URL. Preferably, this product reference URL specifies a request for a secure HTTP transaction. If a secure transaction 56 can be established with the server specified by the product URL, the client browser will autonomously determine whether a cookie having a matching domain and path reference exists on the client system 12. In accordance with the present invention, such a cookie will be marked secure, to prevent transmission in unsecure transactions, and provided with a vendor defined expiration date, to force re-establishment of a credit relationship for inactive client users. If a cookie is found with a matching domain and path 58 the client browser adds the corresponding cookie data to the HTTP URL request message 60 that then is issued to the URL specified server 62. If a cookie is not found or has expired, the URL request message without a cookie is sent to the URL specified server 62.

Detailed Description Text (33):
A secure HTTP transaction will typically not be specified by the first URL of a redirection URL. Consequently, the redirection URL is issued to the sponsor server and a redirection message and second URL will be returned to the client browser 54. Preferably, the second URL specifies a secure transaction request and a server that can support secure transactions.

Detailed Description Text (34):
Alternately, the URL referenced server may refuse or fail the negotiation for a secure transaction with the client browser. In this case, the purchase transaction must be refused or proceed to be handled by the client browser and server in a conventional unsecure manner 57. However, since any cookie stored by the client browser for the referenced server and path is marked secure, the cookie is not sent by the client browser as part of any unsecure HTTP transaction with the URL referenced server.

Detailed Description Text (35):
In accordance with the present invention, the a secure URL referencing a product for purchase is preferably of one of two forms. The first form provides for the execution of a CGI program on the server system 16 to implement the server side operation of the present invention. An exemplary URL conforming to the present invention could be:

Detailed Description Text (36):
In this example, "vendor.com" establishes the domain relative to any client side cookies stored by the client system 12 while the path component "cgi-bin/buy" establishes a base cookie path sufficient to uniquely identify a cookie in combination with the provided domain. The "<productID00357>" information provides the CGI program with the required identification of the product selected for purchase.

Detailed Description Text (38):
Again, for purposes of identifying a client side stored cookie, the "vendor.com" domain and the "catalog/buy" path component are sufficient to uniquely identify a cookie. Preferably, the key word "purchase" is recognized by the HTTPd server application as specifying performance of the secure purchase transaction in accordance with the present invention with respect to the specified "<productID00357>".

Detailed Description Text (39):
Implementation of the present invention through a CGI program external to at least an otherwise conventional HTTPd server application or as a direct internal modification of a generally conventional HTTPd server application does not alter the basic method of performing the present invention. However, each approach to implementation has distinct trade-offs. Execution of external CGI programs can increase the effective load handled by the server system 16. However, the CGI program is typically portable among any particular implementation of the HTTPd server application, assuming the server natively supports for the secure socket layer or an equivalent.

Detailed Description Text (41):
Independent of whether the secure purchase algorithm of the present invention is implemented as an external CGI program or internal modification of the server application itself, the initial substantive action by the HTTPd server application is to determine whether a URL received in a secure transaction with a client browser references a purchasable product 64. The initial parsing of the URL by the HTTPd server application will determine whether a secure purchase transaction CGI program is to be executed by the HTTPd server or whether the "purchase" key word is embedded in the URL. Where neither effective product reference exists in the URL, the URL is further processed by the HTTPd server application in a conventional manner typically for the purpose of serving a Web page to the client browser 66.

Detailed Description Text (42):
Where a product is referenced by the URL, a determination is then made as to whether a cookie is provided as part of the HTTP message providing the received URL 68. Where no cookie is associated with the URL, the submission of the URL is taken to imply a request to establish a purchasing arrangement with the vendor server. Thus, the server system 16 replies to the URL by returning a new account form to the client browser 70. This form may request whatever information is deemed appropriate by the vendor in order to establish an open purchase arrangement with the vendor. Typically, the information requested includes a name, password, credit card number, type of credit card, expiration date, and credit card billing address.

Detailed Description Text (43):
Once the client user has filled out the form, the client browser submits the form

contents to the server system 16 for evaluation 72. The information provided by the client user may be automatically verified with the applicable credit card issuer or agent 74 to determine whether a valid account may be established for the client user. Should this credit check fail, the server system 16 preferably provides a form response to the client browser refusing the purchase transaction 76. Where the credit check succeeds, the server system 16 preferably opens a new account record for the client user 78. In addition, the server system 16 creates an initial cookie that encodes at least a client user identification code (ID) and the password submitted in connection with the new account form 72.

Detailed Description Text (44):
In an alternate preferred embodiment of the present invention, the cookie generated 78 not only encodes a client user ID and password, but also encodes other identifying information that is sent by the client browser as part of the URL request message. Encoding this additional information can serve to uniquely or at least substantially associate the cookie with a specific combination of the client browser and client system 12. In addition, the cookie may be further encrypted utilizing any conventional private key encryption algorithm. The private key and other information utilized in the construction of the cookie is stored with the account record for the client user. The substantive contents of the cookie is not decodable anywhere outside of the server system 16. Consequently, copies of the cookies as stored by the client system 12 are essentially non-portable among other possible client systems or modifiable to allow client user impersonation.

Detailed Description Text (45):
Where a cookie is provided 68 with the URL request issued by the client browser, the cookie is utilized to perform a database look-up to identify a client user account record. Regardless of whether the cookie has been encrypted, the cookie data can be utilized as the account reference for performing the database look-up of a client user record. The cookie data can then be validated against the information present in the account record. Where the cookie has been encrypted, the cookie can be decrypted utilizing the private key present in the account record and then validated against not only the other information stored within the account record but also present identifying information received directly from the client browser as part of the current HTTP transaction.

Detailed Description Text (46):
Where the cookies is determined invalid to due a failure to locate a current and valid account record corresponding to the cookie or in authenticating the cookie, a new account form can be provided to the client browser to establish or re-establish a credit relationship between the client user and vendor.

Detailed Description Text (47):
Where an account record is found and the cookie is authenticated 80, or where a new account has been successfully created 78, the server system 16 preferably then sends a confirmation form to the client browser 82. A new cookie, generated either in connection with the creation of an account record 78 or, where the cookie encodes a generational identifier that inherently changes over time and is potentially sequence specific, following validation of the received cookie 80, is provided by the server system 16 to the client browser in connection with the return of the confirmation form to the client browser. Even if a new cookie is not generated, a cookie is nonetheless preferably still set through an HTTP response to the client browser to update the expiration date associated with the cookie as held by the client browser. In all events, the cookie is marked secure.

Detailed Description Text (48):
In response, the client browser preferably displays the confirmation form on the client system 12 while recording the new or updated cookie in the client side storage provided by the client browser application 84. In a preferred embodiment of the present invention, the confirmation form provides an order summary sufficient

to permit the client user to determine whether to confirm or cancel the order. The client user completes and submits the form by selecting either to accept or cancel the order 86. Where, in an alternate embodiment, the confirmation form provides for accepting, cancelling and deferring purchase of individual products summarized on the confirmation form, the client user completes these choices prior to submitting the form as accepted as annotated or cancelled in its entirety. Assuming that the order is confirmed or at least accepted as annotated, the server operates from the confirmation data provided by the submitted form to update the client user's account record 87 and process the order generally in a conventional manner to provide for the delivery of the selected product to the client user 88.

Detailed Description Text (52):
Thus, a system and method for providing secure, convenient and efficient purchase transactions to be executed over the Internet has been described. The system and method incorporate a substantial degree of flexibility allowing a product or service to be purchased with no more than two required mouse clicks where a credit relationship has been pre-established between a client user and vendor. The flexibility afforded by the present invention in performing purchase transaction extends also to allowing vendors to operate as catalog servers, URL sponsors, proxies for other servers providing purchasable products and services, and to provide purchase clearinghouse activities for the benefit of client users. Consequently, an efficient and convenient system for facilitating purchases over the Internet while maintaining an extremely high degree of security over sensitive information has been described.

CLAIMS:

2. The method of claim 1 wherein said persistent predetermined identifier is encoded with information that allows secure selection of said persistent predetermined identifier in connection with a defined set of URLs including said predetermined URL.

3. A method of claim 2 wherein said persistent predetermined identifier is stored in a secure manner by said client browser and communicated to said merchant server only through a secure network communications transaction.

4. The method of claim 3 wherein said persistent predetermined identifier is stored in an encrypted state by said client browser and includes data specific to said client browser so as to secure said predetermined identifier to said client browser.

6. A method of performing trans-Internet purchase transactions between client browsers and merchant vendors, said method comprising the steps of:

a) providing for a predetermined Web page to be served to a client browser with said predetermined Web page identifying a purchasable item and including a corresponding purchase transaction URL;

b) receiving said corresponding purchase transaction URL and predetermined persistent cookie data previously stored by a merchant vendor on said client browser where said predetermined persistent cookie data is selected by said corresponding purchase transaction URL;

c) determining the identity of said purchasable item and from said corresponding purchase transaction URL; and

d) securely authenticating said client browser based on said predetermined persistent cookie data.

11. A method of presenting an electronic catalogue of purchasable items to a client

browser wherein at least one merchant vendor is represented in the electronic catalogue, said method comprising the steps of:

a) serving a Web page of an electronic catalogue to a client browser, said Web page including identifications of a plurality of purchasable items, each one of said plurality of purchasable item having an associated URL embedded in said Web page;

b) receiving, in response to a single client browser selection, a predetermined URL request from said client browser including predetermined client environment data including predetermined persistent cookie data specifically corresponding to said predetermined URL;

c) validating said client browser and identifying a predetermined one of said plurality of purchasable items from said predetermined URL; and

d) serving a confirmation form to said client browser that requires a maximum of a single selection to accept and conclude the purchase of said predetermined one of said plurality of purchasable items.

12. The method of claim 11 wherein the Web page server is or is acting as an agent or proxy for a merchant vendor, said method further comprising the steps of:

a) identifying said client browser as representing a new account with respect to a merchant vendor;

b) establishing a credit relationship with said client browser on half of said merchant vendor; and

c) storing predetermined persistent cookie data, encoded to identify said client browser to said merchant vendor, on said client browser.

16. A method of enabling purchase transactions for individual items from a plurality of merchant vendors through a common Web page, said method comprising the steps of:

a) embedding a first URL, associated with a purchasable item, in a Web page that is served from a first server to a client browser, said first URL referencing a second server;

b) providing for the storage of first persistent cookie data by said client browser and of a first database record by said second server, said first persistent cookie data corresponding to said first database record as stored by said second server;

c) receiving by said second server a first URL request corresponding to the client browser selection of said first URL;

d) receiving by said second server said first persistent cookie data;

e) validating said first persistent cookie data against said first database record;

f) identifying said purchasable item from said first URL request;

g) obtaining confirmation of the purchase of said purchasable item from said client browser; and

h) providing for said client browser to issue a second URL request to said first server to be served with said Web page.

18. The method of claim 16 further comprising the steps of:

h e b b g e e e f c e e ge

a) said second server providing a third URL request to a third server;

b) providing for the storage of second persistent cookie data by said second server and of a second database record by said third server, said second persistent cookie data corresponding to said second database record as stored by said third server;

c) receiving by said third server said third URL request;

d) receiving by said third server said second persistent cookie data;

e) validating said second persistent cookie data against said second database record;

f) identifying by said third server said purchasable item from said third URL request; and

g) obtaining confirmation of the purchase of said purchasable item from said second server.

21. The method of claim 20 wherein said step of providing for said client browser to issue a second URL request to said first server includes providing new first persistent cookie data to said client browser for storage, said second server storing a new first database record corresponding to said new first persistent cookie data.

22. The method of claim 21 wherein at least a part of said first persistent cookie data and said new first persistent cookie data is encrypted and wherein a corresponding decryption key is stored in said first database record and said second database record.